Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claim 1. (original) A processor, comprising:

a processing core that generates memory addresses to access a main memory and on which a plurality of methods operate, each method using its own set of local variables; and

a data cache subsystem comprising a multi-way set associative cache and a data memory that holds a contiguous block of memory defined by an address stored in a register, wherein local variables are stored in said data memory.

Claim 2. (original) The processor of claim 1 wherein the data memory includes a plurality of lines and a valid bit and a dirty bit associated with each line, and wherein upon completion of a method, the local variables associated with said completed method continue to be marked as valid and not copied back to a main memory even though the lines in which the completed method's local variables are stored are marked as valid and dirty.

Claim 3. (original) The processor of claim 1 wherein when a new method is called, the local variables associated with the called method use data memory space previously used by local variables associated with completed methods without generating a miss.

Claim 4. (original) The processor of claim 1 wherein the data memory includes a plurality of lines, each line being marked as either valid or invalid, and when a line is marked as invalid and data memory hit/miss occurs, a targeted line is

marked as valid causing a line fetch from external memory to occur, wherein a hit/miss indicates that a targeted address hits in the data memory but targeted data is not stored in the data memory.

Claim 5. (original) The processor of claim 1 wherein the data memory includes a plurality of lines, each line being marked as either valid or invalid, and when a line is marked as invalid and data memory hit/miss occurs, a targeted line is marked as valid without causing a line fetch from external memory to occur, wherein a hit/miss indicates that a targeted address hits in the data memory but targeted data is not stored in the data memory.

Claim 6. (currently amended) The processor of claim 1 wherein the data memory containing the local variables has higher priority during hit/miss determinations ~~than all other ways~~.

Claim 7. (previously presented) The processor of claim 1 wherein, if said data memory way storing said local variables does not have sufficient capacity to store the local variables, then at least some local variables are stored in the [2-] multi-way set associative cache.

Claim 8. (original) The processor of claim 1 including a global valid bit that indicates whether the local variables stored in the data memory is valid or not and a valid bit for each of a plurality of entries associated with the data memory indicating whether each entry holds a valid local variable.

Claim 9. (cancelled)

Claim 10. (currently amended) ~~The cache subsystem of claim 9~~ A cache subsystem, comprising:

a multi-way set associative cache;

a data memory that holds a contiguous block of memory defined by an address stored in a register, wherein local variables are stored in said data memory; and

wherein the data memory includes a plurality of lines and a valid bit and a dirty bit associated with each line, and wherein upon completion of a method, the local variables associated with said completed method continue to be marked as valid and not copied back to a main memory even though the lines in which the completed method's local variables are stored are marked as valid and dirty.

Claim 11. (currently amended) The cache subsystem of claim ~~9~~ 10 wherein when a new method is called, the local variables associated with the called method use data memory space previously used by local variables associated with completed methods without generating a miss.

Claim 12. (currently amended) ~~The cache subsystem of claim 9~~ A cache subsystem, comprising:

a multi-way set associative cache;

a data memory that holds a contiguous block of memory defined by an address stored in a register, wherein local variables are stored in said data memory; and

wherein the data memory includes a plurality of lines, each line being marked as either valid or invalid, and when a line is marked as invalid and data memory hit/miss occurs, a targeted line is marked as valid causing a line fetch from external memory to occur, wherein a hit/miss indicates that a targeted address hits in the data memory but targeted data is not stored in the data memory.

Claim 13. (currently amended) ~~The cache subsystem of claim 9~~ A cache subsystem, comprising:

a multi-way set associative cache;

a data memory that holds a contiguous block of memory defined by an

address stored in a register, wherein local variables are stored in said data memory; and

wherein the data memory includes a plurality of lines, each line being marked as either valid or invalid, and when a line is marked as invalid and data memory hit/miss occurs, a targeted line is marked as valid without causing a line fetch from external memory to occur, wherein a hit/miss indicates that a targeted address hits in the data memory but targeted data is not stored in the data memory.

Claim 14. (currently amended) The cache subsystem of claim ~~9~~ 13 wherein the data memory containg the local variables has higher priority during hit/miss determinations.

Claim 15. (currently amended) The cache subsystem of claim ~~9~~ 13 wherein, if said data memory does not have sufficient capacity to store the local variables, then at least some local variables are stored in the multi-way set associative cache.

Claim 16. (currently amended) The cache subsystem of claim ~~15~~ 13 wherein said local variables comprise local variables used in a stack-based instruction set.

Claims 17 – 20. (cancelled)

Claim 21. (currently amended) ~~The method of claim 20~~ A method, comprising the steps of:

programming a register to define a contiguous block of memory in a cache subsystem;

storing local variables associated with executing methods in the contiguous block of memory; and

~~further comprising~~ marking lines in the block as valid and completing a

method while continuing to mark the lines as valid and not copying the local variables associated with the completed method to a main memory.

Claim 22. (cancelled)

Claim 23. (currently amended) ~~The method of claim 20 further comprising~~ A method, comprising the steps of:

programming a register to define a contiguous block of memory in a cache subsystem;

storing local variables associated with executing methods in the contiguous block of memory; and

configuring a global valid bit to indicate whether the local variables are collectively valid and configuring a valid bit for each of a plurality of entries in the block to indicate whether each entry contains valid data.